Secretaria de Educação e Esportes

GOVERNO DO ESTADO Pernambuco

Secretaria Executiva de Desenvolvimento da Educação – SEDE

Gerência de Políticas Educacionais do Ensino Médio – GEPEM

ROBÓTICA LIVRE COM ARDUINO

Recife - 2022

GOVERNADOR DE PERNAMBUCO
Paulo Henrique Saraiva Câmara

SECRETÁRIO DE EDUCAÇÃO E ESPORTES
Marcelo Andrade Bezerra Barros

SECRETÁRIA EXECUTIVA DE DESENVOLVIMENTO DA EDUCAÇÃO
Ana Coelho Vieira Selva

GERÊNCIA GERAL DO ENSINO MÉDIO E ANOS FINAIS DO ENSINO FUNDAMENTAL
Regina Celi de Melo André

GERENTE DE POLÍTICAS EDUCACIONAIS DO ENSINO MÉDIO
Ana Carolina Ferreira de Araújo

EQUIPE DE ROBÓTICA LIVRE COM ARDUINO
José Altenis dos Santos
Gabriel Pimenta Carneiro Campelo

SUMÁRIO

APRESENTAÇÃO 04
INTRODUÇÃO 05
MEDIDA DE TENSÃO DE PILHAS E BATERIAS 07
MEDIDA DE RESISTÊNCIA ELÉTRICA DE RESISTORES 07
CÓDIGO DE CORES DE RESISTORES 08
PROTOBOARD 09
PLATAFORMA ARDUINO 09
PORTAS DIGITAIS 10
PORTAS ANALÓGICAS 10
PORTAS PWM 11
IDE ARDUINO 11
PISCA COM LED 11
POTENCIÔMETRO CONTROLANDO LED 13
CONTROLANDO LED COM SINAL PWM 14
SENSOR DE TEMPERATURA LM35 15
LDR – RESISTÊNCIA DEPENDENTE DE LUZ 16
SENSOR ULTRASÔNICO HC-SR04 17
DS18B20 – SENSOR DIGITAL DE TEMPERATURA 18
SS49E - SENSOR DE CAMPO MAGNÉTICO 20
SHIELD DE LCD 16X2 COM TECLADO 21
BUZZER 23
SERVO MOTOR 25
BLUETOOTH 26
SEGUIDOR DE LINHA TCRT5000 28
RELÉ 29
CONTROLE REMOTO INFRAVERMELHO 30
SENSOR PIR 32
COMPOSIÇÃO DE KIT ROBÓTICA LIVRE COM ARDUINO 34
1º KIT DE EXPANSÃO 37
BIBLIOGRAFIA 39

APRESENTAÇÃO

A Secretaria de Educação do Estado de Pernambuco, atendendo às exigências de uma formação sintonizada com o desenvolvimento do conhecimento, da criatividade e da autoria, traz o Projeto ROBÓTICA LIVRE COM ARDUINO para a escola estadual de ensino médio.

Este projeto de robótica se articula fortemente aos componentes curriculares de Física e Matemática, mas também com todos os outros, garantindo que conteúdos pedagógicos sejam abordados por meio de uma prática interativa e estimulante. Ainda possibilita ao estudante a compreensão de programação, conhecimento bastante importante frente aos desafios da sociedade atual.

Destacamos que o trabalho com robótica na rede estadual vem tendo resultados muito positivos e que trabalhar com a robótica livre irá expandir as possibilidades de desenvolvimento de projetos por parte dos estudantes, ampliando suas possibilidades de compreensão de conceitos, mas também de intervenção na realidade.

Professores terão possibilidade de explorar e ampliar os conhecimentos trazidos neste material, que é um subsídio para o início do projeto nas escolas. Temos certeza que será um projeto valioso e estimulante para professores e estudantes do ensino médio de Pernambuco.

Ana Selva

Secretária Executiva de Desenvolvimento da Educação

INTRODUÇÃO

As novas exigências da vida em sociedade apontam para uma educação que precisa promover um cidadão com uma formação em ciências que esteja conectada ao seu dia a dia, aos seus anseios e ao uso sustentável do meio ambiente. Neste sentido propomos um projeto que utilize a robótica como meio de construir e compreender tecnologias presentes no nosso cotidiano bem como usufruir de forma sustentável nosso ambiente.

O Projeto Robótica Livre visa desenvolver junto às comunidades escolares o uso de robótica numa plataforma cujo software e hardwares são de uso livre tornando-se assim mais acessível. O uso de uma plataforma livre permite acesso a uma quantidade maior de sensores e atuadores, aumentando em muito as possibilidades de projetos, além de ser muito mais acessível tanto na aquisição quanto financeiramente.

Conectada a todas essas ideias, a plataforma escolhida é a do Arduino, que foi desenvolvida no ano de 2005 na Itália com a finalidade de ajudar estudantes de design a desenvolver projetos. Esta plataforma tem a vantagem de já ter uma vasta gama de comunidades na internet que disponibilizam projetos, além de ser de fácil aquisição no comércio.

Este projeto tem uma ampla gama de aplicações promovendo a transversalidade em várias disciplinas como Física, Química, Matemática, Biologia, Geografia ente outras. Os estudantes irão desenvolver aplicações práticas de conceitos estudados teoricamente que envolverão diversas áreas de estudo.

José Altenis dos Santos

ATENÇÃO

Todos os trabalhos descritos neste material são eletricamente alimentados pela ***saída USB*** de um computador ou por uma ***bateria de 9V***.

NÃO USE A REDE ELÉTRICA DE ALTA TENSÃO (220V) NESTES PROJETOS.

Medida de Tensão de Pilhas e Baterias

O multímetro é um aparelho utilizado para realizar medidas de tensão, corrente e resistência elétrica. Na nossa primeira atividade vamos utilizá-lo para medir tensão elétrica em uma bateria. Devemos ligar a ponteira vermelha do multímetro **DT33C** em **VΩmA** e a ponteira preta em **COM** e o cursor do multímetro em **20V** tensão contínua. Consulte o manual do multímetro para maiores detalhes. Deve-se então colocar a ponteira vermelha no positivo da bateria e a ponteira preta no negativo.

Exercício:

Agora anote o valor da tensão indicado no corpo da bateria e o valor da tensão medido com o multímetro nesta mesma bateria **em seu caderno** como no modelo da tabela abaixo.

Valor escrito no corpo da bateria	
Valor medido com o multímetro	

Medida de Resistência Elétrica de Resistores

Devemos ligar a ponteira vermelha do multímetro **DT33C** em **VΩmA** e a ponteira preta em **COM** e o cursor do multímetro em **Ω** resistência elétrica num valor acima do valor do resistor. Por exemplo, se o resistor for de 1kΩ a primeira e melhor opção é colocar a chave do multímetro em 2000Ω. Consulte o manual do multímetro para maiores detalhes. Deve-se então colocar o a ponteira vermelha num terminal do resistor e a ponteira preta no outro terminal, com o resistor desconectado de qualquer circuito.

Código de Cores de Resistores

No código de cores de resistores temos fundamentalmente 4 cores. As três primeiras cores indicam o valor do resistor e a quarta cor a possível variação em porcentagem deste valor.

1ª cor – algarismo ; 2ª cor – algarismo; 3ª cor – potência multiplicadora; 4ª cor – porcentagem de variação

Cor	Dígitos Significativos (1 e 2)	Multiplicador (3)	Tolerância (4)
Preto	0	10^0	
Marrom	1	10^1	± 1%
Vermelho	2	10^2	± 2%
Laranja	3	10^3	
Amarelo	4	10^4	
Verde	5	10^5	
Azul	6	10^6	
Violeta	7	10^7	
Cinza	8	10^8	
Branco	9	10^9	
Ouro	-	10^{-1}	± 5%
Prata	-	10^{-2}	± 10%
Sem Cor	-	-	± 20%

Exemplo: Qual seria o valor do resistor abaixo? Faça as contas no seu caderno.

Exercício

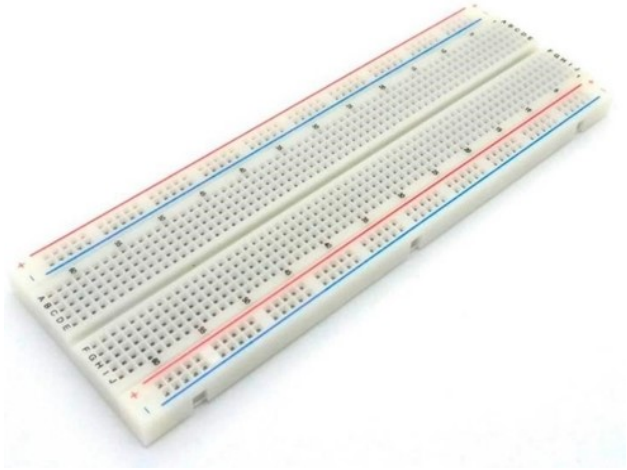
Vamos agora a um exercício prático. Pegue um resistor no laboratório e preencha **no seu caderno** as tabelas no modelo abaixo e compare o valor medido com o valor indicado pelas cores.

	1ª cor Algarismo	2ª cor Algarismo	3ª cor Potência	4ª cor Tolerância
Cores				
Valor da cor				
Resultado				

Valor medido com multímetro	

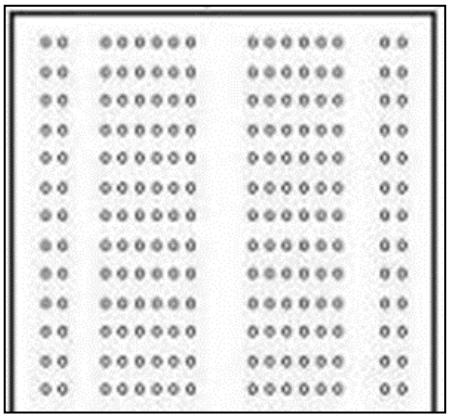
Protoboard

A protoboard é uma placa para montar circuitos de protótipos. Ela tem ligações elétricas internas que você deve compreender para poder montar os circuitos. Utilizaremos uma protoboard de 830 pontos .



Exemplo:

A imagem abaixo representa o esquema de uma protoboard sem as ligações internas. Reproduza essa imagem **em seu caderno** e depois adicione as ligações elétricas internas da protoboard.



Plataforma Arduino

O Arduino é um sistema formado de software e hardware livre criado em 2005 por professores universitários na Itália. Esse sistema foi criado com o intuito com que pessoas leigas em eletrônica e programação tivessem a vida facilitada na construção de projetos e por certo atingiu seu intuito, pois o Arduino, hoje, é amplamente utilizado por leigos e também por especialistas.

Uma das principais vantagens desse sistema é a grande quantidade de projetos prontos e a disposição na internet para serem acessados. Gerou-se em torno dessa plataforma um grande movimento de criação, criatividade, interatividade.

Outra grande vantagem do uso da plataforma Arduino é a facilidade de encontrar peças no mercado, além do preço acessível dos componentes.

Portas Digitais

O Arduino Uno tem 14 portas digitais nomeadas de 0 a 13, que podem ser de entrada ou saída. As portas 0 e 1 são utilizadas para comunicação serial e devem ser evitadas nas programações. Uma porta digital deve assumir somente dois níveis 0 (zero) ou 1(um) que no Arduino que trabalha com tensões entre 0 e 5V assumem em termos de tensões 0V (nível baixo) ou 5V (nível alto).

Portas Analógicas

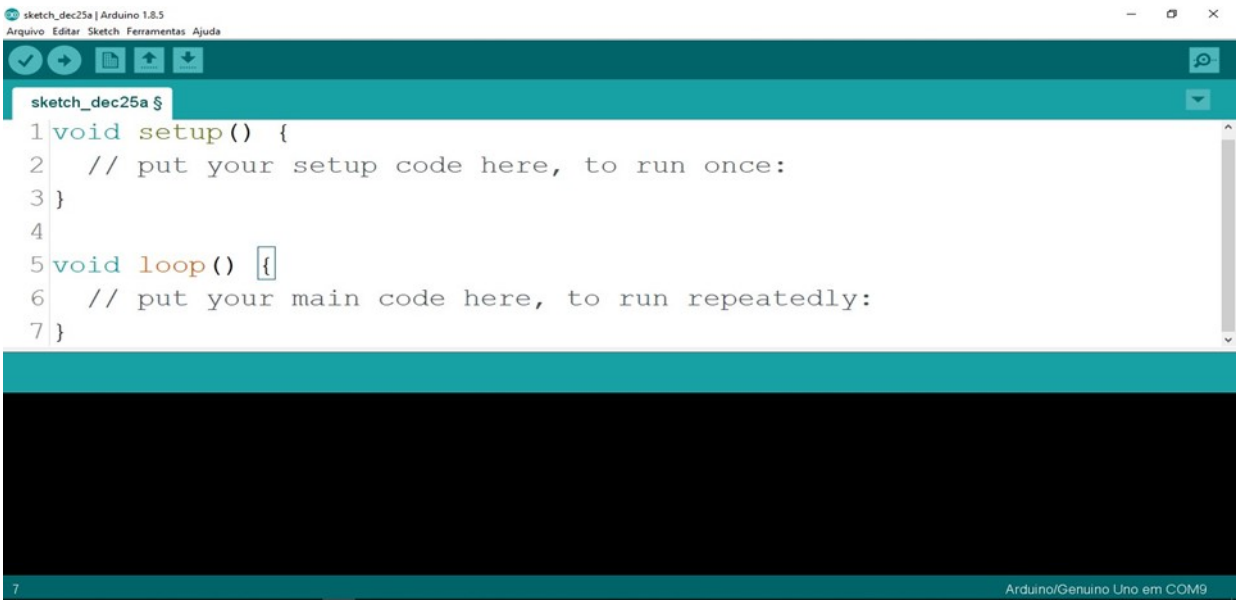
O Arduino Uno tem 6 portas analógicas nomeadas de A0 a A5 e são apenas de entrada. As portas analógicas podem receber sinais com tensões entre 0 e 5V e através de um conversor analógico digital (ADC) converter em um byte com 10 bits.

Portas PWM

O Arduino Uno tem 6 portas digitais do tipo PWM: são elas 3,5,6,9,10 e 11. Estas portas podem modular a largura do pulso digital e com isso conseguem simular uma saída analógica entre 0 e 5V.

IDE Arduino

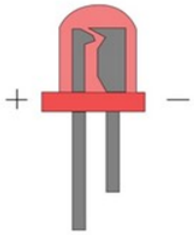
O ambiente de programação no Arduino (IDE) é muito semelhante a linguagem C. A medida que as montagens forem sendo realizadas vamos aprendendo como usar esse ambiente de programação. Os arquivos de programação no Arduino são denominados de sketch.



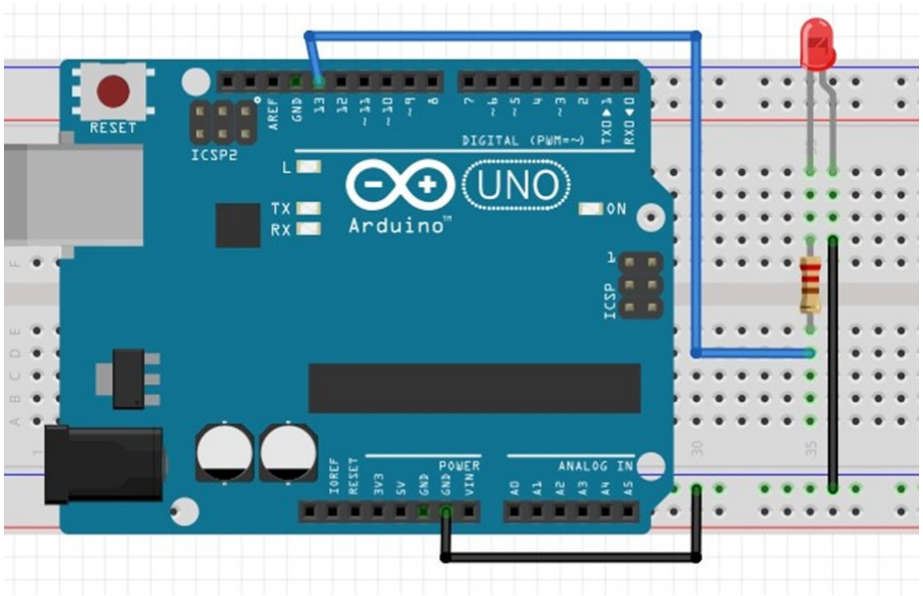
Pisca com LED

Neste primeiro exemplo será utilizada uma porta digital como saída para controle de um LED. Com o advento dos semicondutores, a eletrônica moderna passou a produzir Diodos Emissores de Luz (LED). Este componente tem um baixíssimo consumo de corrente elétrica sendo muito versátil para ser agregado a vários projetos diferentes.

Neste roteiro vamos aplicar sobre ele um sinal digital que aciona e desliga o LED com um intervalo de tempo predeterminado. Como o LED é um diodo, possui polaridade. Ou seja, um dos terminais (maior comprimento/cátodo) deve receber o polo positivo e o outro (menor com primento/ânodo) fica acionado na porta comum (GND).



Esquema de Ligações na Protoboard:



Programação:

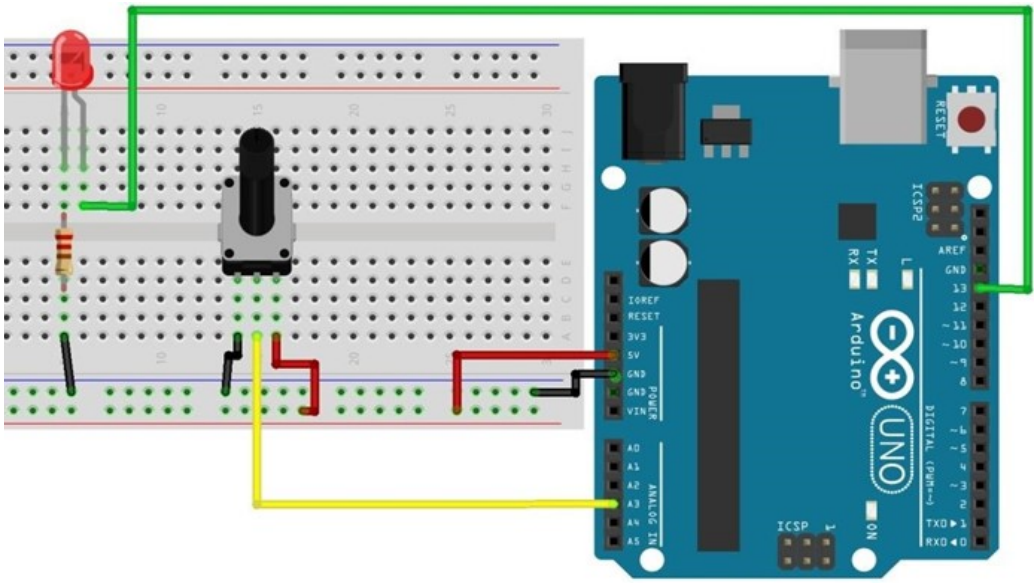
```
int LED = 13;                    //declaração de uma variável chamada LED

void setup() {                   //função de ajuste executada uma única vez
pinMode (LED, OUTPUT);           //rotulando a variável com saída de dados
}

void loop() {
digitalWrite(LED, HIGH);         //torna variável LED ligada -HIGH - tensão máxima  5 V
delay(1000);                     //espera um segundo (1000 milisegundos)
digitalWrite(LED, LOW);          //torna variável LED desligada -LOW - tensão mínima 0V
delay(1000);
}
```

Potenciômetro Controlando LED

O potenciômetro é um resistor variável cuja resistência pode ser variada através de um eixo central. O potenciômetro do kit tem valor de 1MΩ e é do tipo B, ou seja, tem resposta linear com o eixo central.

Entre os dois terminais extremos (1 e 3) a resistência é de 1MΩ, podendo variar entre 0 e 1MΩ entre o terminal central (2) e um dos terminais extremos. Neste experimento o potenciômetro funciona como controle e utilizamos uma porta analógica.

Esquema de Ligações na Protoboard:



Programação:

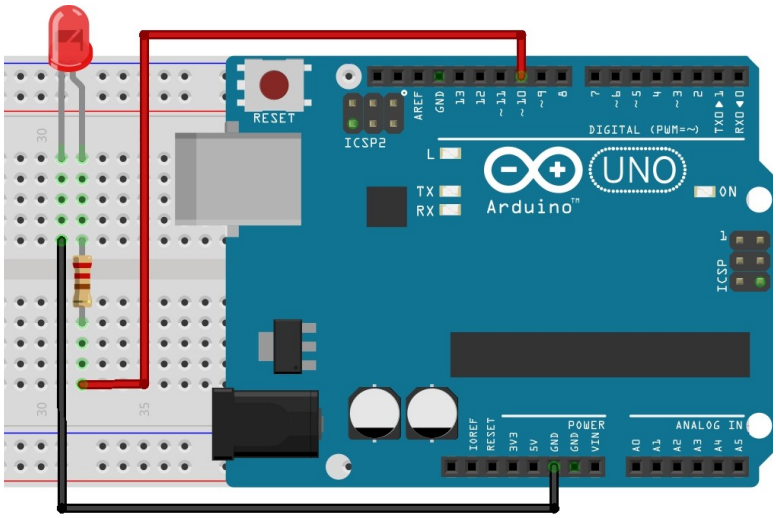
```
int sensorPin = A3;                    // seleciona o pino de entrada do potenciômetro
int ledPin = 13;                       // seleciona o pino do LED
int sensorValue =0;                    //variável para guardar o valor vindo do sensor

void setup() {
Serial.begin(9600);                     // inicia comunicação com serial
pinMode(ledPin, OUTPUT);                // declara ledPin como uma saída
}
```

```
void loop()
{
sensorValue = analogRead(sensorPin);      // lê o valor do sensor
Serial.println(sensorValue);              // imprime sensorValue na serial
digitalWrite(ledPin, HIGH);               // liga o LED
delay(sensorValue);                       // interrompe por <sensorValue> milisegundos
digitalWrite(ledPin, LOW);                // desliga o LED
delay(sensorValue);                       //interrompe por <sensorValue> milissegundos
}
```

Controlando LED com Sinal PWM

Neste projeto vamos aplicar sobre o LED um sinal digital que varia com o tempo (PWM), com isso observaremos a potência luminosa variar linearmente criando um efeito de pulsação. A saída PWM irá variar de 0 a 255 e isto fará a saída variar entre 0 e 5V.

Esquema de Ligações na Protoboard:



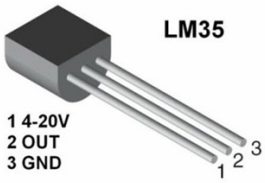
Programação:

```
int led=10;                               //Cria variável led e faz igual a 10
int brilho=0;                             //Cria variável brilho e faz igual a 0
int pulo=5;                               //Cria variável pulo e faz igual a 5

void setup ()
{                                         //função de configuração
pinMode (led, OUTPUT);                    //Coloca pino led=10 no modo saída
}
```

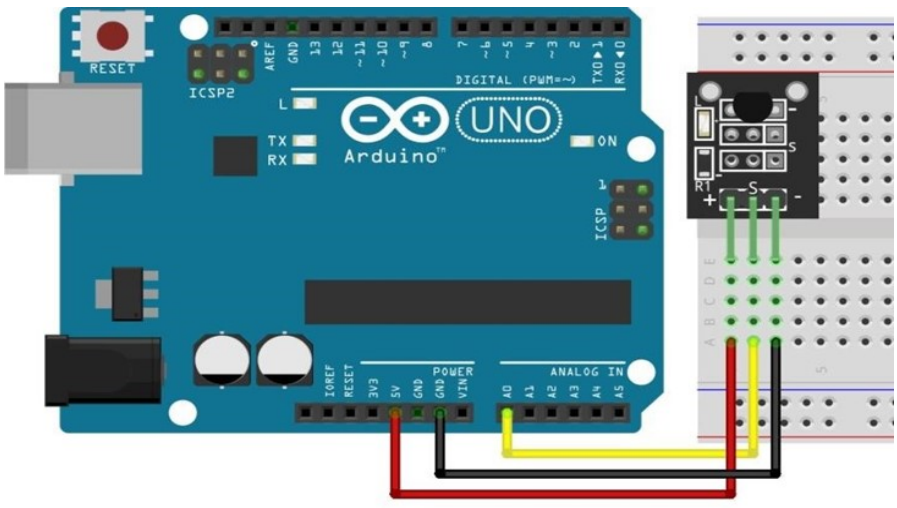
```
void loop ()
{                                          //função repetição
analogWrite (led, brilho);                 //Escreve no pino led=10 o valor do brilho
brilho = brilho + pulo;                    //Faz brilho igual a soma de brilho e pulo
if (brilho==0 || brilho==255) {            //Se brilho=0 ou brilho=255 (||=ou)
pulo= - pulo;                              //pulo igual a menos pulo
}
delay(30);                                 //esperar 30 milisegundos
}
```

Sensor de Temperatura LM35

Este componente é um sensor de temperatura linear com uma precisão de 0,5°C cuja faixa de medição está entre –55°C e 150°C e sua tensão de saída varia de 10mV/°C. A alimentação do LM35 deve ser com tensões entre 4 e 20V e tem consumo de corrente muito baixo na faixa de 60µA, estando assim na faixa de uso do Arduino.



Esquema de Ligações na Protoboard:



Programação:

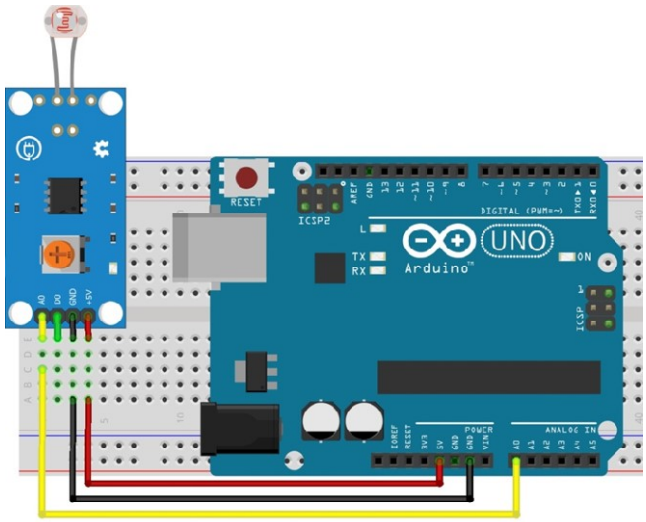
```
const int LM35 = A0;                          // Define o pino que lera a saída do LM35
float temperatura;                            // Variável que armazenará a temperatura medida

void setup() {
Serial.begin(9600);                           //Inicia comunicação com a serial
}

void loop() {
//fórmula que transforma tensão em temperatura
temperatura = (float(analogRead(LM35))*5/(1023))/0.01;
Serial.print("Temperatura (Celsius): ");      //Imprimi o texto "Entre Aspas"
Serial.println(temperatura);                  //Imprimi a variável temperatura
delay(2000);                                  //Espera 2000 milisegundos -  2 segundos
}
```

LDR - Resistência Dependente de Luz

Trata-se de um resistor que varia seu valor de acordo com a intensidade da luz que incide sobre o sensor. Comumente utilizado em sistemas de controle de luminosidade para acendimento automático de luzes.

Esquema de Ligações na Protoboard:

Neste esquema é usado um módulo com 4 terminais: GND, VCC(+5V), A0 E D0. O terminal A0 é o analógico e o terminal D0 o digital, que pode ser regulado pelo trimpot.



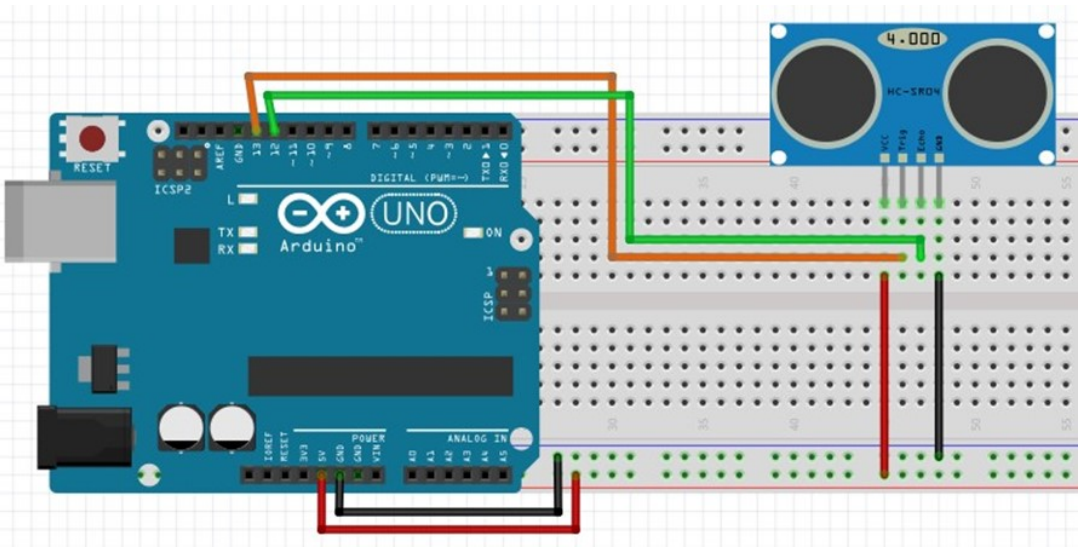
Programação:

```
int LDR = 0;                          //declara a variável LDR
int val = 0;                          //declara a variável val

void setup() {
Serial.begin (9600);                  //inicia o monitor serial
}

void loop() {
val = analogRead (LDR);               //adquire o valor na porta analógica e atribui a "val"
Serial.println (val);                 //exibi em tela o valor da variável "val"
delay (1000);                         //espera 1000 milisegundos = 1 segundo
}
```

Sensor Ultrasônico HC - SR04

A finalidade deste sensor é aferir a distância entre dois pontos. Para isso ele utiliza um sinal ultrassonoro (fora do alcance dos ouvidos humanos) cronometrando o tempo entre a emissão e recepção do sinal junto com o valor da velocidade do som, pode-se determinar a posição da barreira em relação ao sensor.

Esquema de Ligações na Protoboard:



Programação:

```
int trigPin = 13;                     //declaração da variável de emissão
int echoPin = 12;                     //declaração da variável de recepção
```

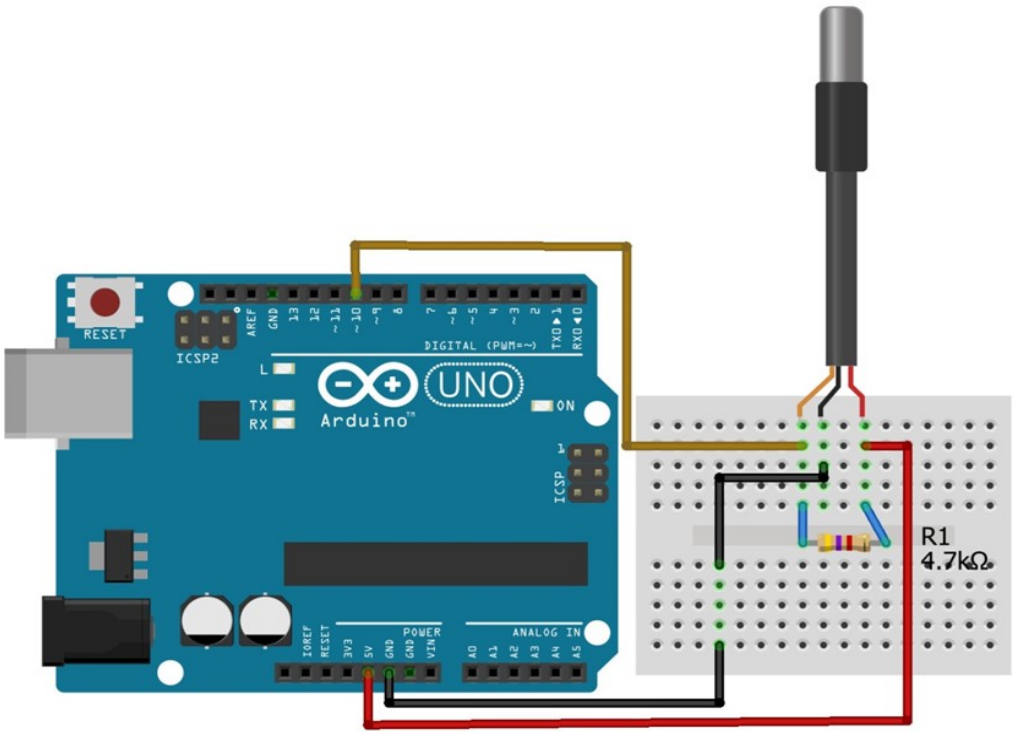
```
void setup()
{
Serial.begin (9600);                    //declara e regula a velocidade de troca de dados
pinMode (trigPin, OUTPUT);              //rotula a variável como saida de dados
pinMode (echoPin, INPUT);               //rotula a variável como entrada de dados
}

void loop()
{
long duration, distance;                //declara as variáveis duration e distance
digitalWrite (echoPin, LOW);            //inicia o receptor de sinal como desligado
delayMicroseconds (2);                  //aguarda 2 microssegundo
digitalWrite (trigPin, HIGH);           //aciona o emissor de sinal
delayMicroseconds (10);                 //aguarda 10 microssegundos
digitalWrite (trigPin, LOW);            //desliga o emissor de sinal
duration = pulseIn (echoPin, HIGH);     //conta o tempo entre a emissão e recepção do sinal
distance = (duration / 2)*0.034;        //equação que transforma tempo em distância
Serial.println(distance);               //Envia distance a serial
delay(1000);                            //Espera 1 segundo
}
```

DS18B20 - Sensor Digital de Temperatura

O DS18B20 é um sensor digital de temperatura. O DS18B20 tem comunicação serial e funciona entre temperaturas de -55°C e 125°C. Cada DS18B20 tem um código serial de 64 bits que permite vários deles utilizarem a mesma linha serial para se comunicar com o microprocessador.

Esquema de Ligações na Protoboard:

Observe que entre os terminais positivo e data do DS18B20 deve-se usar um resistor de 4k7Ω.



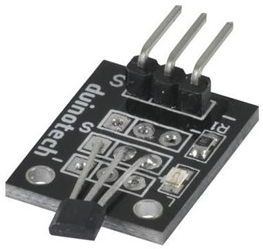
Programação:

Nesta programação utilizamos a biblioteca "DallasTemperature", que é um programa, sub-rotinas, criado pelo fabricante do sensor para facilitar seu uso, sua programação. Observe que começamos a programação chamando esta biblioteca com o camando "#include <DallasTemperature.h>". Fique atento que esta biblioteca deve estar instalada na IDE Arduino.

```
#include <DallasTemperature.h>                  //Chama biblioteca DallasTemperature
#define DS18B20 10                              //Define pino 10 comunicação do DS18B20
OneWire ourWire(DS18B20);                       //Escolhe a comunicação do DS18B20
DallasTemperature sensors(&ourWire);            //Escolhe a variável sensors

void setup() {
sensors.begin();                                // Inicia o sensor DS18B20
}

void loop() {
Serial.begin(9600);                             //Inicia comunicação com a serial
sensors.requestTemperatures();                  //Requisita a temperatura do sensor
Serial.println(sensors.getTempCByIndex(0));     //Envia temperarura a serial
delay(1000);                                    //Faz uma espera de 1 segundo
}
```

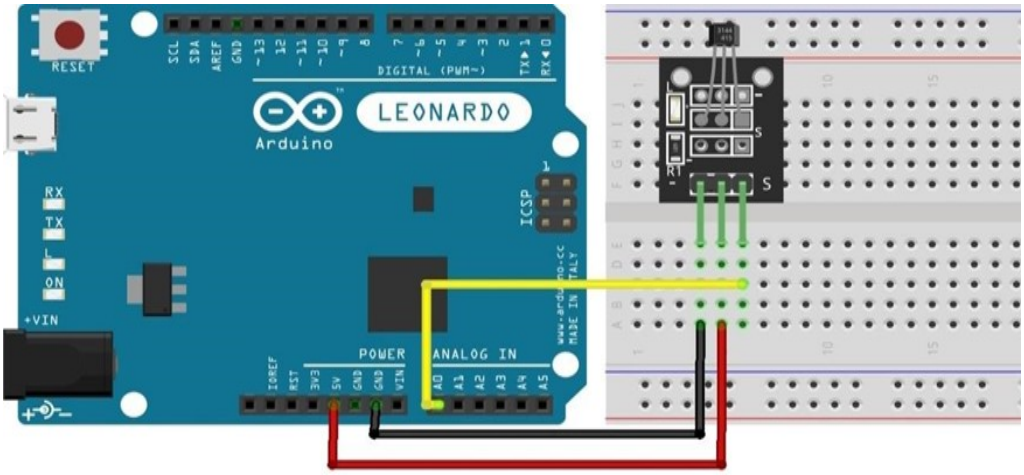
SS49E - Sensor de Campo Magnético

O sensor hall 49e é um sensor linear que quando é atravessado por um campo magnético produz uma corrente elétrica proporcional ao campo magnético.



O sensor 49e mede campos magnéticos entre -1500 Gauss e +1500 Gauss e produz tensões na saída que vão de 0,86V a 4,21.

Esquema de Ligações na Protoboard:



Programação:

```
float refVoltage = 5.0/1024;                    //Cria variável de referência
float sensorVolts;                              //Cria variável de tensão
int val;                                        //Cria variável de valor

void setup()
{
Serial.begin(9600);                             //Inicia comunicação serial
}
```

Programação:

```
void loop()
{
val=analogRead(0);                                  //Ler a porta A0 e coloca em val
sensorVolts=refVoltage*(val);                       //Calcula valor da tensão
Serial.print("B=");                                 //Imprime o texto "B=" na serial

// Imprime na serial o valor de B já calculado
Serial.print((sensorVolts*895.52773449)-2238.8193362 );
Serial.println("Gauss");                            //Imprime na serial a unidade de B

//Define se o campo magnético é do tipo Norte ou Sul e imprime na serial
if (((sensorVolts*667)-1667) > 2) { Serial.println ("  Norte");};
if ( ((sensorVolts*667)-1667) < -2 ) {Serial.println ("  Sul");};
delay (1000);                                       //Esperar 1 segundo
}
```

Shield de LCD 16x2 com Teclado

Este shield encaixa-se sobre o Arduino, logo neste projeto não haverá esquema de ligações na protoboard. O shield tem 2 linhas de dígitos, cada uma com 16 dígitos. Além disso possui um teclado para movimentação e escolha em menus.

Na programação deste shield devemos evitar utilizar as portas digitais 4 a 10 e a analógica A0, pois estas já são utilizadas pela biblioteca “LiquidCrystal.h”.

Deve-se saber também que a porta analógica A0 é utilizada para os 5 botões, assim tem-se o botão Direita(0 - 99), Esquerda (400 - 599), Cima(100 - 199), Baixo(200 - 399), Select(600 - 799) e existe também um sexto botão que serve como reset para o Arduino. Também existe parafuso no trimpot para ajuste do contraste do display.

Programação:

Observe que nesta programação devemos usar a biblioteca “LiquidCrystal.h” e a primeira linha de programação é exatamente chamando esta subrotina.

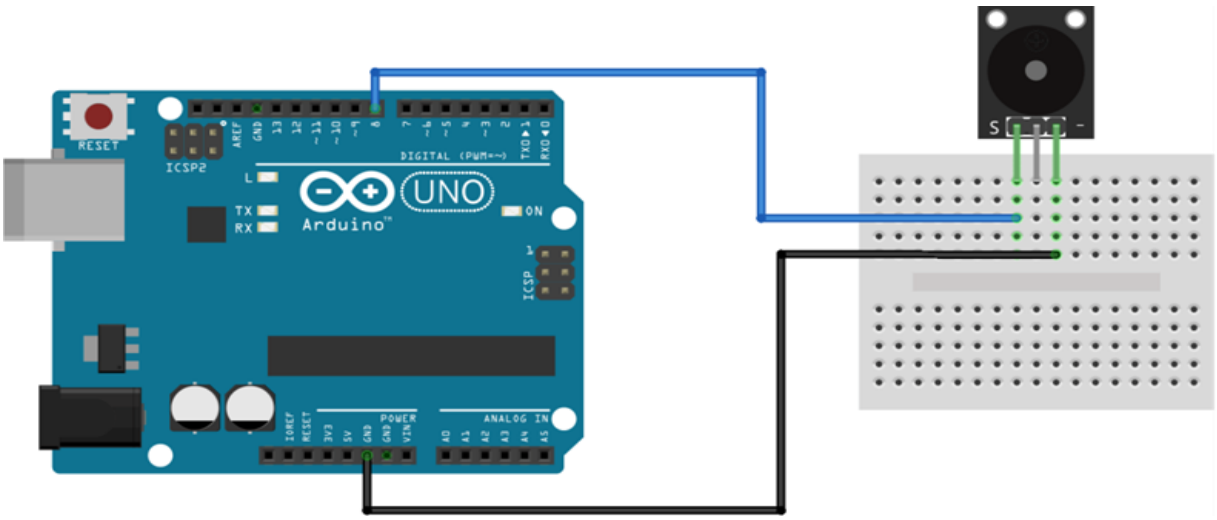
```
#include <LiquidCrystal.h>                  //Chama a biblioteca LiquidCrystal
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);        //portas usadas pelo shield

void setup()
{
lcd.begin(16, 2);                           //iniciando lcd 16x2
lcd.setCursor(0,0);                         // colocar cursor na posição (0,0)
lcd.print("Teste de Shield");               //mensagem na posição (0,0)
lcd.setCursor(0,1);                         // colocar curso na posição (0,1)
lcd.print("Tecla :");                       //mensagem na posição (0,1)
}

void loop()
{
int botao;                                  //cria variável botao
botao = analogRead (0);                     //Leitura do valor da porta analógica A0
lcd.setCursor(8,1);                         //coloca cursor na posição (8,1)

if (botao < 100) {                          //Se a botao menor que 100
lcd.print ("Direita");                      //Escreva Direita
}
else if (botao < 200) {                     //Se a botao menor que 200
lcd.print ("Cima");                         //Escreva Cima
}
}
```

```
else if (botao < 400){                    //Se a botao menor que 400
lcd.print ("Baixo   ");                   //Escreva Baixo
}
else if (botao < 600){                    //Se a botao menor que 600
cd.print ("Esquerda");                    //Escreva Esquerda
}
else if (botao < 800){                    //Se a botao menor que 800
lcd.print ("Select  ");                   //Escreva Select
}
}
```

Buzzer

O buzzer passivo é um atuador que emite sons. Para utilizá-lo usamos a função **"tone"** que tem o seguinte formato: *tone (pino,* frequência*,* duração*)* onde a frequência do tom é em hertz, e a duração em mili segundos.

Esquema de Ligações na Protoboard:



Programação:

```
#define tempo 10                          //Define tempo de atuação do comando tone
int frequencia = 0;                       //Cria variável frequência
int Pinofalante = 8;                      //Define pino do buzzer

void setup()
{
Serial.begin(115200);                     //Inicia comunicação com serial
pinMode(Pinofalante, OUTPUT);             //Pino do buzzer
}

void loop()
{
//Varia frequencia entre 150 e 1800 somando 5
for (frequencia = 150; frequencia < 1800; frequencia += 5)
{
tone(Pinofalante, frequencia, tempo);     //Emite som
Serial.println(frequencia);               //Imprime na serial
delay(1);                                 //Espera 1 milisegundo
}
//Varia frequencia entre 1800 e 150 subtraindo 5
for (frequencia = 1800; frequencia > 150; frequencia -= 5)
{
tone(Pinofalante, frequencia, tempo);     //Emite som
Serial.println(frequencia);               //Imprime na serial
delay(1);                                 //Espera 1 milisegundo
}
}
```

Secretaria de Educação e Esportes

GOVERNO DO ESTADO Pernambuco

Secretaria Executiva de Desenvolvimento da Educação – SEDE

Gerência de Políticas Educacionais do Ensino Médio – GEPEM

ROBÓTICA LIVRE COM ARDUINO

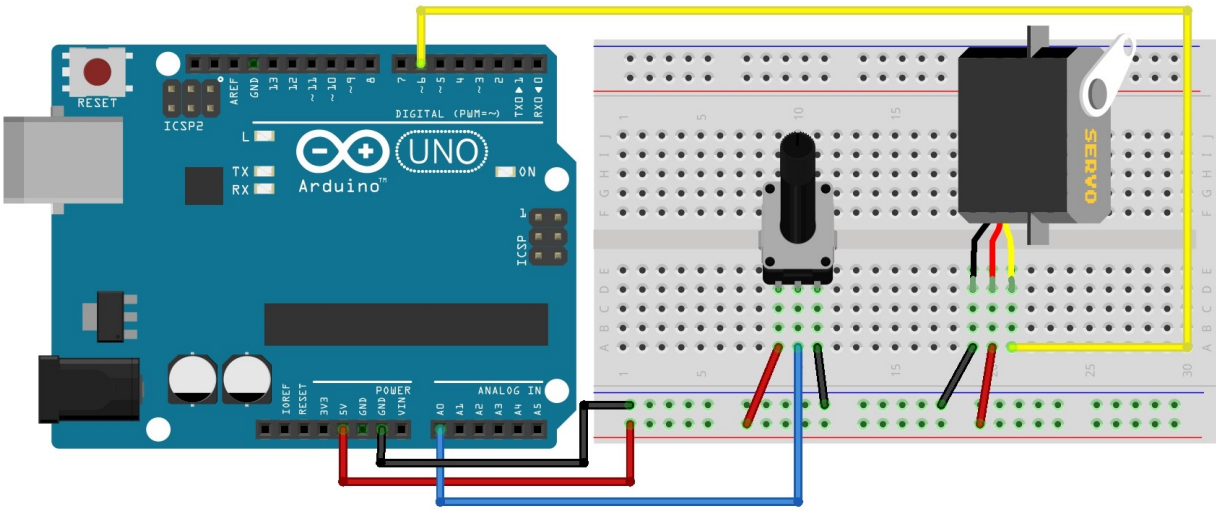
1º Kit de Expansão

Recife - 2022

Servo Motor

O servo motor é um atuador de alta precisão que pode girar entre 0° e 180° . No projeto a seguir um potenciômetro irá atuar no controle do ângulo do servo motor.

Esquema de Ligações na Protoboard:



Programação:

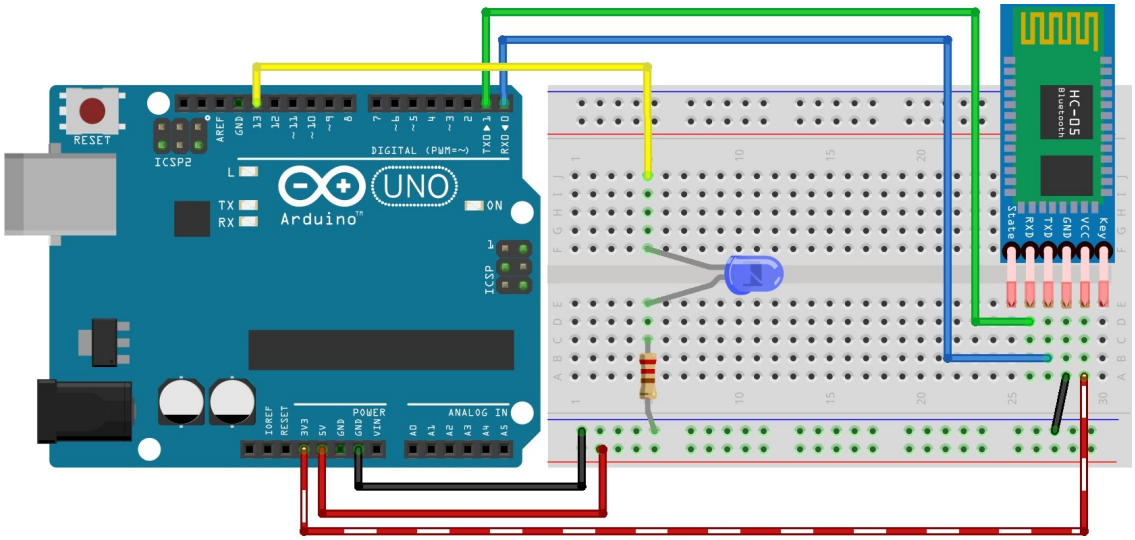
Observe que nesta programação devemos usar a biblioteca “Servo.h” e a primeira linha de programação é exatamente chamando esta biblioteca. Fique atento que esta biblioteca deve estar instalada na IDE Arduino.

```
#include <Servo.h>
Servo servo1;                          //Cria o servo1

void setup() {
servo1.attach(6);                      //servo1 ficará na porta D6
}

void loop(){
int angle = analogRead(0);             //Cria variável angle e lê Potenciômetro na porta A0
angle=map(angle,0,1023,0,180);         //Mapeia o valor de 0 a 1023 para o de 0 a 180 graus
servo1.write(angle);                   //Escreve o ângulo em servo1
delay(15);                             //Tempo de 15ms para o servo1 alcançar a posição
}
```

Bluetooth

O bluetooth é uma tecnologia de comunicação entre vários dispositivos e tem como características o baixo uso de energia, baixa taxa de transmissão de dados (1Mbit/s), um alcance de 10m e trabalha na frequência de 2.4GHz. Neste projeto um smartphone com o APP **Multi-ControlBT** irá se comunicar com um Bluetooth que está conectado ao Arduino para controlar um led.

Esquema de Ligações na Protoboard:

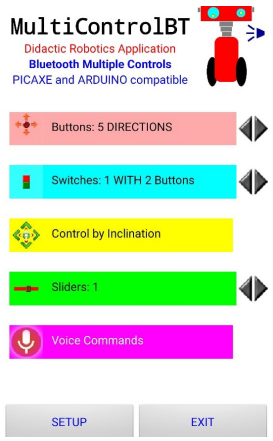


Programação:

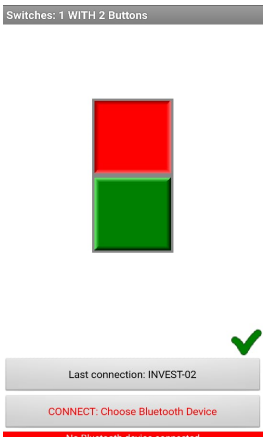
```
int state;                          // Cria variável state

void setup() {
pinMode(13,OUTPUT);                 // Coloca a porta 13 em modo saída
Serial.begin(9600);                 // Inicia a comunicação com a serial
}

void loop() {
if (Serial.available()>0){          // Verifica se existe dado disponível na serial
state=Serial.read();                // Lê a serial e coloca na variável state
if (state==5) {                     // Ajuste o APP do celular para enviar 5 para desligar o led
digitalWrite(13, LOW);              // Deixa a porta 13 baixa desligando o led
}
if (state==8) {                     // Ajuste o APP do celular par enviar 8 para ligar o led
digitalWrite(13,HIGH);              // Deixa a porta 13 alta ligando o led
}
}
}
```

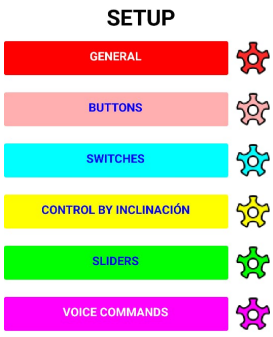
APP MultiControlBT:

Depois de instalar o APP MultiControlBT escolha a opção "**Switches**" e ajuste para **1 WITH 2 Buttons** e no setup do SWITCHES ajuste para enviar 5 e 8. Na verdade esta já é a configuração básica.

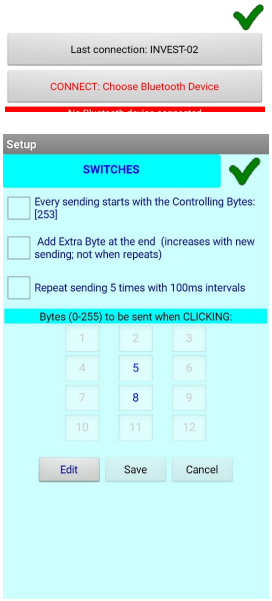


Esta é a tela inicial. Escolha a opção **Switches** e aparecerá a tela ao lado direito que permitirá conectar o bluetooth e controlar o led.

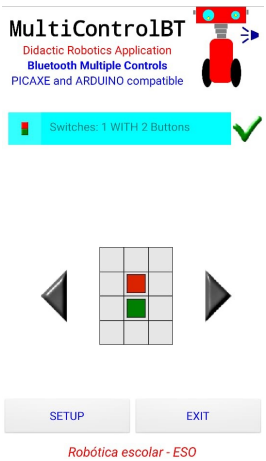




Na tela inicial, escolha o botão **SETUP** que aparecerá a tela a esquerda. Agora escolha o **ícone configurar** em **Switches** que permitirá configurar os dados enviados na tela a direita. Clique em Edit para editar os números. Neste caso já está em 5 e 8.



Na tela **inicial** escolha **as setas** ao lado do botão **Switches** que permitirá configurar quantos conjuntos de botões você irá utilizar. Isto para o caso de querer controlar 2 ou 3 leds, por exemplo.

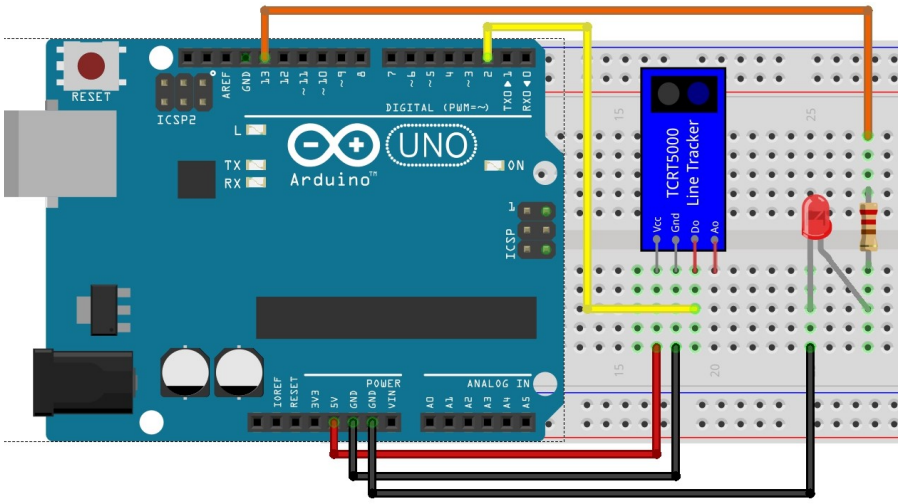


Não esqueça de ativar o bluetooth do seu celular e detectar o bluetooth que está no Arduino. Para isto o bluetooth deve estar ligado e piscando rapidamente. Reconheça o bluetooth do Arduino no seu celular, ele irá pedir uma senha (pin). Quando você conseguir, o LED que piscava rapidamente irá mudar para outra condição, ficando somente ligado ou piscando lentamente.

Seguidor de Linha TCRT5000

O módulo TCRT5000 é composto de um emissor infravermelho e um receptor. A partir da quantidade de luz refletida em uma superfície consegue-se identificar uma faixa na cor preta em uma superfície branca ou vice-versa.

Esquema de Ligações na Protoboard:

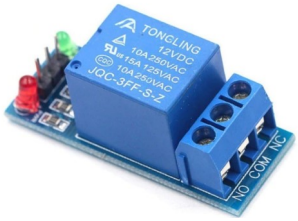


Programação:

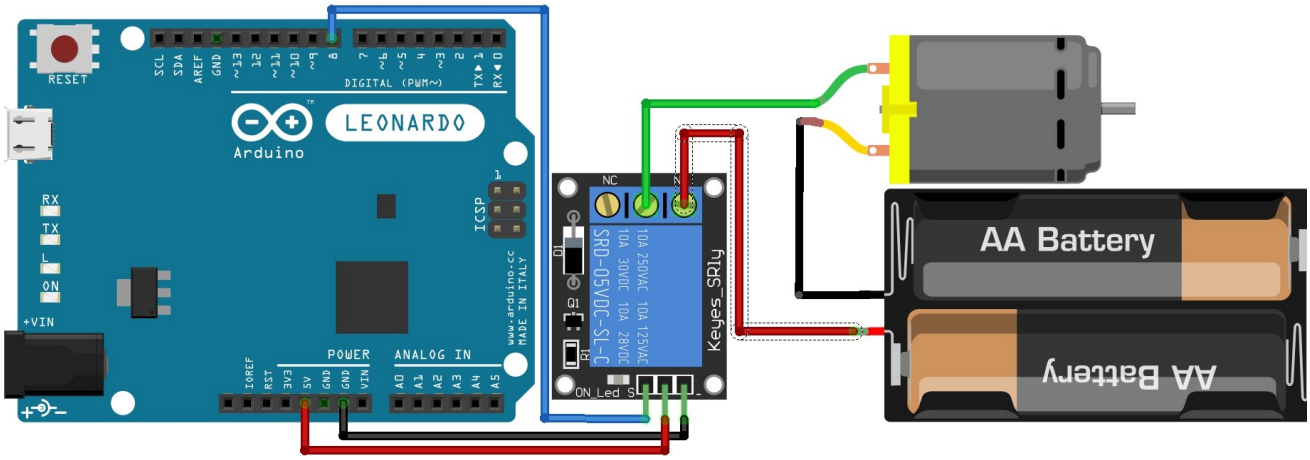
```
int ledPin = 13;                    // Cria variável ledPin igual a 13 para led
int inPin = 2;                      // Cria variável inPin para o TCRT5000
int valor = 0;                      //Cria variável e atribui valor zero

void setup() {
pinMode(ledPin, OUTPUT);            // Coloca o pino 13 como output
pinMode(inPin, INPUT);              // Coloca o pino 2 como input
}

void loop() {
valor = digitalRead(inPin);         // Lê o valor do pino 2 e coloca em "valor"
if (valor == HIGH) {                // Se valor input for HIGH
digitalWrite(ledPin, HIGH);         // Ligar o LED
}
else {                              // Se não
digitalWrite(ledPin, LOW);          // Desligar o LED
}
}
```

Módulo Relé

O Módulo Relé é uma chave eletromecânica que quando é energizada aciona um contato fechando um circuito. Esse funcionamento permite que um circuito isolado acione outro de potência maior.



Esquema de Ligações na Protoboard:



Programação:

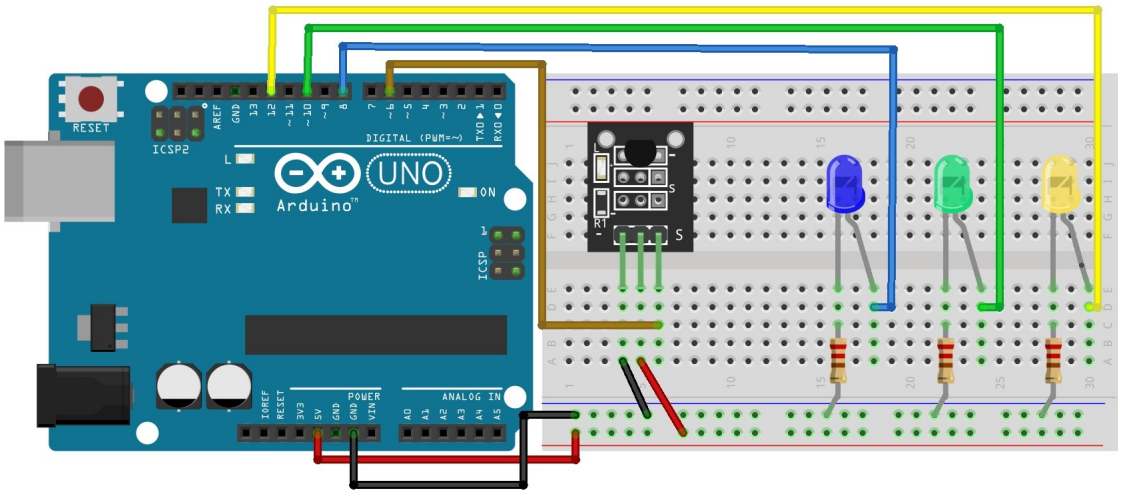
Será utilizado o teclado no monitor serial para ligar (L) e desligar (D) o módulo relé.

```
const int RelePin = 8;                          // pino ao qual do Relé
int LigaDesliga;                                // variavel para ler dados teclado

void setup() {
Serial.begin(9600);                             // inicia a comunicação serial em 9600bps
pinMode(RelePin, OUTPUT);                       // define o pino como saída
}

void loop() {
if (Serial.available() > 0) {                   // verifica dados na serial
LigaDesliga = Serial.read();                    //guarda o dado em LigaDesliga

if (LigaDesliga == 'L') {                       //se LigaDesliga for L
digitalWrite(RelePin, HIGH);                    //aciona o pino
}

if (LigaDesliga == 'D') {                       //se for LigaDesliga for D
digitalWrite(RelePin, LOW);                     //desativa o pino
}
}
}
```

Controle Remoto Infravermelho

O controle remoto emite um sinal infravermelho em uma determinada frequência, neste caso 38KHz, que tem um código diferente para cada tecla. Do outro lado tem-se um receptor infravermelhor configurado para trabalhar também na frequência de 38KHZ, este receptor está ligado ao Arduino que decodifica o sinal enviado.

Esquema de Ligações na Protoboard:



Programação:

```
#include <IRremote.h>                  //Inclui a biblioteca IRremote.h

int IR_PIN = 6;                        //Cria variável IR_PIN para pino do infravermelho
float armazenavalor;                   //Variável que armazenará valor do sinal infravermelho
int pinledazul = 8;                    //Led Azul no pino 8
int pinledverde = 10;                  //Led Verde no pino 10
int pinledamarelo = 12;                //Led Amarelo no pino 12
IRrecv irrecv(IR_PIN);                 //Define pino do receptor de infravermelho
decode_results results;

void setup()
{
pinMode(pinledazul, OUTPUT);           //Torna o pino do led azul saída
pinMode(pinledverde, OUTPUT);          //Torna o pino do led verde saída
pinMode(pinledamarelo, OUTPUT);        //Torna o pino do led amarelo saída
Serial.begin(9600);                    //Inicia comunicação com a serial
irrecv.enableIRIn();                   // Inicializa o receptor Infravermelho
}
```

Observe que os valores hexadecimais das teclas estão em **negrito** pra destacar que os valores podem ser diferentes dependendo do tipo de controle remoto. Assim verifique no monitor serial o valor hexadecimal das teclas, quando digitadas, e modifique o código do sketche, com os valores corretos, se necessário.

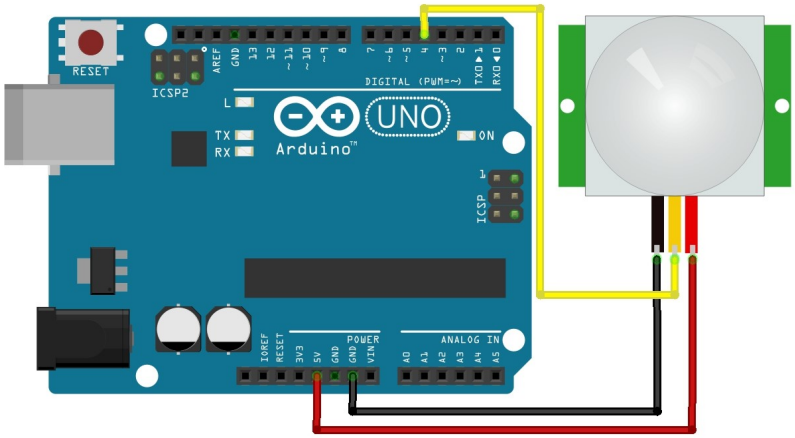
```
void loop()
{
if (irrecv.decode(&results))
{
Serial.print("Valor lido : ");
Serial.println(results.value, HEX);          //Imprime valor hexadecimal no monitor serial
armazenavalor = (results.value);
if (armazenavalor == 0xFF30CF)               //Verifica se a tecla 1 foi acionada
{
digitalWrite(pinledazul, HIGH);              //Acende o led azul
}
if (armazenavalor == 0xFF18E7)               //Verifica se a tecla 2 foi acionada
{
digitalWrite(pinledazul, LOW);               //Apaga o led azul
}
if (armazenavalor == 0xFF10EF)               //Verifica se a tecla 4 foi acionada
{
digitalWrite(pinledverde, HIGH);             //Acende o led verde
}
if (armazenavalor == 0xFF38C7)               //Verifica se a tecla 5 foi acionada
{
digitalWrite(pinledverde, LOW);              //Apaga o led verde
}
if (armazenavalor == 0xFF42BD)               //Verifica se a tecla 7 foi acionada
{
digitalWrite(pinledamarelo, HIGH);           //Acende o led amarelo
}
if (armazenavalor == 0xFF4AB5)               //Verifica se a tecla 8 foi acionada
{
digitalWrite(pinledamarelo, LOW);            //Apaga o led amarelo
}
if (armazenavalor == 0xFF52AD)               //Verifica se a tecla 9 foi acionada
{
digitalWrite(pinledazul, LOW);               //Apaga led azul
digitalWrite(pinledverde, LOW);              //Apaga led verde
digitalWrite(pinledamarelo, LOW);            //Apaga led amarelo
}
if (armazenavalor == 0xFF6897)               //Verifica se a tecla 0 foi acionada
{
digitalWrite(pinledazul, HIGH);              //Acende led azul
digitalWrite(pinledverde, HIGH);             //Acende led verde
digitalWrite(pinledamarelo, HIGH);           //Acende led amarelo
}
irrecv.resume();                             //Lê o próximo valor
}
}
```

Sensor PIR

O sensor HC-SR501 PIR(Passive InfraRed) é um sensor infravermelho que detecta movimento de corpos que emitem radiação infravermelha por volta de 10µm, por exemplo seres humanos e outros animais de corpo quente. Este sensor tem um ângulo de "visão" cônico de 100º e seu alcance pode variar entre 3m e 7m ajustado pelo trimpot à direita.

Quando detecta algum movimento o sensor PIR coloca sua saída em nível alto (3,3V) por um período entre 5s e 300s, que também poder ser ajustado pelo trimpot da esquerda, e quando não há movimento sua saída fica em nível baixo(0V).

Esquema de Ligações na Protoboard:

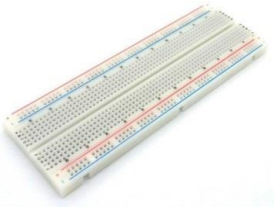


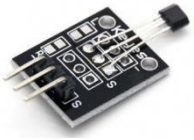
Programação:

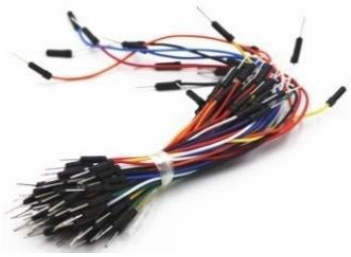
```
int ledPin = 13;                    //Cria variável ledPin e atribui valor 13
int inputPin = 4;                   //Cria variável inputPin e atribui valor 4
int pirState = LOW;                 //Cria variável pirState e atribui valor LOW
int val = 0;                        //Cria variável val e atribui valor 0

void setup() {
pinMode(ledPin, OUTPUT);            //Torna ledPin saída
pinMode(inputPin, INPUT);           //Torna inputPin saída
Serial.begin(9600);                 //Inicia comunicação com a serial
}
```

```
void loop(){
val = digitalRead(inputPin);                          //Lê pino inputPin e atribui a val

if (val == HIGH) {                                    //Se val for HIGH
digitalWrite(ledPin, HIGH);                           //Escreve em ledPin HIGH

if (pirState == LOW) {                                //Se pirState for LOW
Serial.println("Corpo em movimento detectado!");      //Imprime texto “entre aspas”
pirState = HIGH;                                      //Faz pirState HIGH
}
}
else {                                                // Se não
digitalWrite(ledPin, LOW);                            //Escreve em ledPin LOW

if (pirState == HIGH){                                //Se pirState for HIGH
Serial.println("Nenhum corpo em movimento!");         //Imprime texto “entre aspas”
pirState = LOW;                                       //Faz pirState igual a LOW
}
}
}
```

Composição de Kit Robótica Livre com Arduino

Material	Imagem	Quantidade
Arduino Uno **com** cabo		**1**
Protoboard 840 pontos		**1**
Multímetro Digital		**1**
Led 5mm verde		**6**
Led 5mm amarelo		**6**
Led 5mm vermelho		**6**
Resistor 1KΩ 1/4W		**5**
Resistor 47kΩ 1/4W		**5**
Resistor 4k7Ω 1/4W		**5**
Resistor 220Ω 1/4W		**5**
Display Lcd Keypad Shield 16x02 com Teclado		**1**
Trena de Aço 5m		**1**

Módulo Sensor LDR		**1**
Potênciometro 1MΩ com knob		**1**
Sensor Sonda de Temperatura A Prova De Agua D'agua Ds18b20		**1**
Módulo Sensor de Temperatura Lm35		**1**
Módulo Sensor Hall 49E		**1**
Sensor ultrassônico Hc-sr04		**1**
Módulo buzzer passivo		**1**
Chave push-button com apenas 2 terminais		**5**
Mini chave gangorra com apenas 2 terminais		**2**
Bateria de 9V		**1**

Plug para bateria de 9V		**1**
Conector P4 para Arduino		**1**
Conjunto com 60 Jumpers de tamanhos variados		**1**
Alicate de corte pequeno		**1**
Computador para programação do Arduino		**1**
Apostila Robótica Livre com Arduino		**1**

1º Kit de Expansão

Servo Motor - SG90 Mini		**1**
Módulo Bluetooth Serial HC-05		**1**
Controle Remoto Infravermelho 38KHz **+** Módulo receptor infravermelho TSOP 1838		**1**
Módulo Seguidor de Linha TCRT5000		**1**
Módulo Relé 1 Canal 5V		**1**
Módulo Sensor de Presença Sensor PIR		**1**

ATENÇÃO

Todos os trabalhos descritos neste material são eletricamente alimentados pela *saída USB* de um computador ou por uma *bateria de 9V*.

NÃO USE A REDE ELÉTRICA DE ALTA TENSÃO (220V) NESTES PROJETOS.

BIBLIOGRAFIA

Arduino. Disponível em: <https://www.arduino.cc/>. Acesso em: Dezembro 2018.

Thingiverse. Disponível em: <https://www.thingiverse.com/>. Acesso em: dez. 2018.

Instructables, 2018. Disponível em: <https://www.instructables.com/>.

EVANS, M.; NOBLE, J.; HOCHENBAUM, J. **Arduino em Ação**. São Paulo: Novatec, 2013.

LAUDARES, F. A. L.; ALBUQUERQUE, B. N. **Instrumentação para o Ensino de Física Baseada na Plataforma Arduino**. 1. ed. Rio de Janeiro: UFRRJ, 2013.

MCROBERTS, M. **Arduino Básico**. São Paulo: Novatec, 2011.

SILVEIRA, J. A.; **Experimentos com o Arduino.** 2. ed. Campinas: [s.n.], 2013.

MONK, S. **Programação com o Arduino**. Porto Alegre: Bookman, 2013.

MONK, S. **30 Projetos com Arduino**. 2. ed. Porto Alegre: Bookman, 2014.

MONK, S. **Programação com o Arduino II**. Porto Alegre: Bookman, 2015.

SANTOS, J. A. **Instrumentação Eletrônica com o Arduino Aplicada ao Ensino de Física**. Garanhuns, p. 69. 2015. Disponível no site: http://www.sib.ufrpe.br/.

Secretaria de
Educação e Esportes

GOVERNO DO ESTADO
Pernambuco